

09-18-00

A

FISH & RICHARDSON P.C.

4350 La Jolla Village Drive
Suite 500
San Diego, California
92122

Telephone
858 678-5070

Facsimile
858 678-5099

Web Site
www.fr.com

September 15, 2000

Attorney Docket No.: 10559/350001/P9864

Box Patent Application
Commissioner for Patents
Washington, DC 20231

Presented for filing is a new original patent application of:

Applicant: RANDY BONELLA AND JOHN HALBERT

Title: DIGITAL SYSTEM OF ADJUSTING DELAYS ON CIRCUIT
BOARDS

Enclosed are the following papers, including those required to receive a filing date
under 37 CFR 1.53(b):

| | <u>Pages</u> |
|---------------|-------------------------------|
| Specification | 8 |
| Claims | 10 |
| Abstract | 1 |
| Declaration | [To be Filed at a Later Date] |
| Drawing(s) | 3 |

Enclosures:

— Postcard.

There are 44 total claims, 6 of which are independent.

| | |
|--|-----|
| Basic filing fee | \$0 |
| Total claims in excess of 20 times \$18 | \$0 |
| Independent claims in excess of 3 times \$78 | \$0 |
| Fee for multiple dependent claims | \$0 |
| Total filing fee: | \$0 |

CERTIFICATE OF MAILING BY EXPRESS MAIL

Express Mail Label No EL558601705US

I hereby certify that this correspondence is being deposited with the
United States Postal Service as Express Mail Post Office to Addressee
with sufficient postage on the date indicated below and is addressed to
the Commissioner for Patents, Washington, D.C. 20231

September 15, 2000
Date of Deposit

Signature

Derek W. Norwood
Typed or Printed Name of Person Signing Certificate

09/15/00
jc685 U.S. PTO

Federick P. Fish
1855-1930

W.K. Richardson
1859-1951



BOSTON

DALLAS

DELAWARE

NEW YORK

SAN DIEGO

SILICON VALLEY

TWIN CITIES

WASHINGTON, DC

09/15/00 "09/15/00"

jc682 U.S. PTO
09/662054
09/15/00

FISH & RICHARDSON P.C.

Commissioner for Patents
September 15, 2000
Page 2

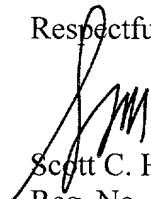
No filing fee is being paid at this time. If this application is found to be incomplete, or if a telephone conference would otherwise be helpful, please call the undersigned at (858) 678-5070.

Kindly acknowledge receipt of this application by returning the enclosed postcard.

Please send all correspondence to:

SCOTT C. HARRIS
Fish & Richardson P.C.
Customer Number: 20985
4350 La Jolla Village Drive, Suite 500
San Diego, CA 92122

Respectfully submitted,


Scott C. Harris
Reg. No. 32,030
Enclosures
SCH/rpi
10053731 doc

005760-091500

APPLICATION
FOR
UNITED STATES LETTERS PATENT

TITLE: DIGITAL SYSTEM OF ADJUSTING DELAYS ON CIRCUIT
BOARDS

APPLICANT: RANDY BONELLA AND JOHN HALBERT

CERTIFICATE OF MAILING BY EXPRESS MAIL

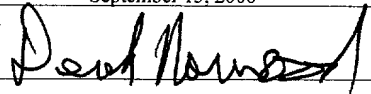
Express Mail Label No. EL558601705US

I hereby certify that this correspondence is being deposited with the United States Postal Service as Express Mail Post Office to Addressee with sufficient postage on the date indicated below and is addressed to the Commissioner for Patents, Washington, D.C. 20231.

September 15, 2000

Date of Deposit

Signature



Derek W. Norwood

Typed or Printed Name of Person Signing Certificate

005760-15029960

DIGITAL SYSTEM OF ADJUSTING DELAYS ON CIRCUIT BOARDS

BACKGROUND

Lengths of lines on a computer circuit board may affect
5 the signals passing through the system. For example, system
buses often run a number of lines in parallel. The lengths of
the lines, however, may cause a delay between the times when
the signals arrive.

In order to minimize the length induced delay between the
10 different elements of the bus, layout considerations have been
used. For example, trial and error techniques may be used to
fine tune the signal paths, to allow signals to arrive in
synchronism.

Extra lengths, such as serpentines, may be added at
15 different areas on the layout.

BRIEF DESCRIPTION OF THE DRAWINGS

The drawings show:

Figure 1 shows a hardware implementation for
20 levelization;

Figure 2 shows a flowchart of levelization; and

Figure 3 shows a flowchart of clock skew operation.

DETAILED DESCRIPTION

The present invention teaches a system which addresses levelization. This system finds application in, for example, computer systems, such as memory interconnects, front side bus
5 interconnect, graphics devices, I/O interconnects, and any other device which uses multiple bit interconnections.

A hardware solution is described which produces specified types of programmable delays in such a system. The hardware solution can include a register which sets the desired delay
10 on each of the plurality of lines.

A first embodiment is shown in Figure 1. An element 100 produces a number of outputs shown as bus 110. The bus 110 can include a plurality of lines, which may or may not have mismatch routing delays. These lines are buffered by I/O
15 buffers 115. Each of the lines 120, 122, 124, 126 is shown in Figure 1. While only four lines are shown in Figure 1, it should be understood that the bus may have many more lines. For example, buses often have 8, 16 or 32 or more lines.

Programmable delay elements 130, 132, 134, 136 are
20 provided on the lines. In this embodiment, there is a single delay element for each line, however, one or more of the lines may be configured without the delay element. Each delay element is controlled by a respective control line 138. In addition, there can be multiple delay lines for each line.

096654-0950
005760-15029960

The signals 120, 122, 124, 126 are delayed by the respective delay elements to form (delayed) output signals 160, 162, 164, 166. These signals are coupled to the core logic 170, which can be circuitry that depends on the function of the device, e.g, memory, graphics, motherboard chipsets or other applications described herein.

A levelization register 140 stores a plurality of values. It can store a single different value for each of the delay elements 130 - 136. These delay values can delay the signals by specified amounts; e.g., by amounts which cause the signals to arrive substantially simultaneously, or within predetermined times of one another.

Arbitration logic 150 carries out the determination of values to be stored in the levelization register. Arbitration logic 150 can be coupled to the output lines 160, 162, 164, 166. The arbitration logic 150 reviews the signals 160 through 166, and determines desired timing information. For example, the arbitration logic 150 may determine whether adjacent lines are leading or trailing each other. Based on this determination, the settings for delay elements 130-136 are determined. These settings cause the values on output lines 160, 162, 164, 166 to be delayed to a specified relationship, e.g., substantially synchronized, or specifically offset.

The information is stored in this levelization register 140. The values are used to delay, bidirectionally, the signals on the signal path.

Alternatively, the arbitration logic 150 can determine leading and lagging edges of signals, and can program those values in the levelization register.

The programmable delay elements 130-136 can be formed using any element which can delay a signal by a variable amount based on an applied signal. These can include a digital delay element, a phase locked loop (PLL) which has multiple taps, a digital locked loop (DLL), or other techniques such as current starving transistors to delay the propagation of the signal therethrough. Moreover, as described above, not each line needs a delay element, e.g., some of the output lines may not include delay elements. However, it may be preferred that each of those lines which includes the delay element includes its own separate delay line 138.

This system can be used in a number of different applications. One application of the present system is in high speed parallel interconnection, where the bus length may actually affect the way that the data arrives at the core logic. This can also be used in memory interconnects, front

side bus interconnect, graphics, IO interconnects and the like.

As described above, the arbitration technique carried out by arbitration logic 150 determines the values that are stored in the levelization register 140, which in turn stores contents for leveling mismatch in the delay elements 130 -136.

Different ways of carrying out the arbitration are disclosed herein.

In general, the arbitration in arbitration logic 150 need only be carried when there is a system event. A system event could include, for example, a change of components installed in the system, such as new memory and/or new cards installed or anything else that might change some issue associated with signal delay. Different kinds of events that should cause new arbitration can be defined. In a Windows™ system, the events can be stored in the system registry and can be triggered by the Plug and Play™ detecting system, for example.

A catastrophic system problem such as a crash, or the like can also cause a new arbitration to be carried out.

The flowchart in Figure 2 shows the arbitration technique. It can be run by a processor, a microcontroller, or can be executed using dedicated hardwired logic, programmed into a field programmable gate array or into discrete circuitry using Hardware Design Language.

The system operates starting in a hard boot at 200 where a special flag is determined at 205. The flag indicates whether a system event of the defined type has occurred. If not, control passes to 210, where levelization values are
5 restored from non volatile memory 142. The information is stored in the levelization register 140, and subsequently used for levelization of signals at 215.

If a system event has occurred at 205, then the system runs through a process of levelizing skews of the
10 interconnects, beginning at 220. Interconnect skew levelization first comprises the master issuing a signal to the "slave" device 100, to cause signals to be produced on the external data bus 110. Each line is dithered by adding a certain amount of delay amount. When all signals arrive at
15 the same time, or within a specified resolution of one another, then the current delays in the levelization register are taken as being the proper levelization delays. This is carried out physically by having checking the alignment of edges at 230. If the edges are not aligned at 235, the
20 arbiter issues a change to levelization register to check the alignment using the next dither value. Flow returns to 225 where the master issues a signal to the slave to again produce signals to produce a new test.

When the edges are aligned at 230, then the current settings are taken as being proper, and the levelization register settings are read and stored in the non volatile memory 142.

5 A special margin test operation is shown as line 250. The margin test operation in more detail in Figure 3. System timing margin is well-known in the art as being the level that is needed to set up and hold lines to the clock edge.

10 However, in this system, programmable delays can be changed under user control. Because of these programmable delays, certain devices operate outside of the normal timing margins. This means that parts of the system are operating at a time that is delayed relative to the operation of other parts of the system.

15 System timing margin can be carried out in the system as shown in Figure 3. Again, the delays in the levelization register 140 are dithered at 300. Each time a different dither occurs, the master issues a signal to the slave at 305, and a data comparator in the core logic 170 checks for failed data. If the system passes, then those values are taken as
20 being usable, and the next value is used. Any failed values are read at 315, compared to the expected margin at 320, and used to form a report. The report 325 is used to set values for the levelization register. These values do not

necessarily cause the signals to arrive simultaneously, but rather cause the signals to arrive in a way that allows some separated objects in the core logic to operate relative to one another, as desired. For example, certain parts may be
5 delayed intentionally relative to other parts for some system timing reason.

Although only a few embodiments have been disclosed in detail above, other modifications are possible. For example, other programmable delay elements could be used.

10

095750-4329960

What is claimed is:

1. A system, comprising:

a data source, having a plurality of different lines;

a plurality of programmable delay elements, each coupled
5 to one of said plurality of lines, to control a delay in said
one of said lines to produce delayed values; and

a register, storing values for said programmable delay
elements which respectively control an amount of delay caused
by said delay elements.

10

2. A system as in claim 1, wherein there are one of
said programmable delay elements for each of said plurality of
lines.

15

3. A system as in claim 1, wherein said register stores
a plurality of values, each of said plurality of values
controlling one of said programmable delay elements.

20

4. A system as in claim 3, further comprising a non
volatile memory, storing said plurality of values.

5. A system as in claim 1, further comprising an
arbitration logic, coupled to said plurality of delayed

values, and operating to determine relative timing of said plurality of lines.

6. A system as in claim 5, wherein said arbitration logic includes a first element which produces a set of first values for said register, and a second element which determines relative arrival of signals based on said first values.

7. A system as in claim 6, wherein said arbitration logic dithers between different sets of values, and determines which of said plurality of values produces best desired result, and stores said best result.

8. A system as in claim 1, further comprising a graphics device, and wherein said signals are from said graphics device.

9. A system as in claim 1, further comprising a non-volatile memory, storing values for said delay elements, and loading said values into said register at a specified time.

10. A system as in claim 8, further comprising arbitration logic, coupled to said plurality of delayed

values, dithers between different sets of values, determines which of said plurality of values produces a best desired result, and stores said best result in said non-volatile memory.

5

11. A system as in claim 10, wherein said best result is one where the plurality of delayed signals are received at substantially the same time.

10

12. A system as in claim 10, wherein said best result is one where the plurality of delayed signals are received at a time skew that allows certain logic elements to operate correctly.

15

13. An electronic device, comprising:

a first device producing a plurality of first outputs;

a plurality of programmable delay elements, each of said plurality of programmable delay elements connected at one end to one of said plurality of outputs and each producing a

20

second output which is delayed relative to said first outputs;

a levelization register, storing a plurality of values, said values each individually controlling one of said delay elements to control an amount of delay caused by said delay element to one of said plurality of outputs.

14. A device as in claim 13, further comprising arbitration logic, connected to each of said second outputs, and determining a relative delay among said second outputs.

5

15. A device as in claim 14, wherein said arbitration logic is responsive to a system event flag, which indicates a specified event in the system.

10 16. A device as in claim 15, wherein said specified event is a hardware change.

17. A device as in claim 15, wherein said specified event is a system crash.

15

18. A system as in claim 15, wherein said arbitration logic is responsive to said flag to produce a first set of values for said levelization register, command said first device to produce said signals, and determine a relative delay among said signals based on said first set of values.

20

19. A system as in claim 14, further comprising a non volatile memory which stores levelization values, said

arbitration logic storing said levelization values in said non volatile memory.

20. A system as in claim 15, further comprising, at
5 initial system start up, downloading values from said non volatile memory to said levelization register.

21. A system as in claim 15, wherein said programmable delay elements are phase locked loops.

10

22. A method comprising:

receiving a plurality of signals from an external device,
each of said plurality of signals related to each other; and

programmably delaying some of said signals relative to
15 others of said signals according to prestored values.

23. A method as in claim 22, wherein plurality of signals are signals from a bus.

20 24. A method as in claim 23, wherein said plurality of signals are signals from a graphics bus.

25. A method as in claim 22, wherein said delaying comprises storing delay values in a non volatile memory; and

using said values in said non volatile memory to adjust a value of a programmable delay element.

25. A method as in claim 24, further comprising
5 determining if a system event has occurred, and storing new delay values in said non volatile memory responsive to said system event occurring.

26. A method as in claim 25, wherein said system event
10 is a change in system hardware configuration.

27. A method as in claim 25, wherein said system event is a system crash.

28. A method as in claim 25, wherein said reobtaining
15 comprises dithering values in a register that stores values for said programmable delay, determining results, and accepting values which have produced a specified delay.

29. A method as in claim 28, further comprising storing
20 said values in said non-volatile memory.

30. A method as in claim 28, wherein said specified delay is a result where there is a minimal delay between arrival of all signals.

5 31. A method as in claim 28, wherein said specified delay is a result where there is a specified delay between arrival of all signals which allows for clock skew in at least one specified logic element.

10 32. A method of equalizing time delays of signals, comprising:

providing a plurality of signals which are produced in times that are synchronized with one another;

15 delaying each of said plurality of signals by a respective amount, wherein each of said respective amounts is different than each other respective amount for a different one of said signals;

testing said signals, to determine relative amounts of delays in said signals, to produce said delay amount; and

20 using said delay amounts to delay said signals.

33. A method as in claim 32, wherein said plurality of signals are signals from a graphics processing device.

34. A method as in claim 32, wherein said delaying comprises delaying each of the signals by respective amounts which causes them to arrive at a specified location at substantially similar times.

5

35. A method as in claim 32, wherein said delaying comprises delaying said signals by specified amounts which causes them to arrive at said location at specified times which are skewed relative to one another, wherein said skew is related to a clock margin of a system.

10

36. A method of setting delays in a system, comprising:
storing values indicative of time delays in a register,
said time delays representing delays to be applied to signals
to obtain a specified result;

15

detecting a system event which indicates that said time delays should be changed;

when said event is not detected, using said values in said register to cause signal delays, by applying said values to respective programmable delay elements; and

20

when said event is detected, using a logic element to determine new delay values and applying said new delay values to said programmable delay elements to cause signal delays based on said new delay values.

005760-15029960

37. A method as in claim 36, wherein said using comprises applying a plurality of delay values to a plurality of respective programmable delay elements to thereby delay a plurality of lines.

5

38. A method as in claim 36, further comprising storing said new delay values in a non-volatile memory.

39. A method as in claim 36, wherein said system event
10 includes a change of system components.

40. A method as in claim 36, wherein said system event is an operating system crash.

15 41. A method, comprising:

receiving a plurality of signals in parallel from a specified device, which signals are produced at substantially synchronized times;

applying said plurality of said signals to programmable
20 delay devices which allow individual delay of said signals;
and

controlling said plurality of programmable delay devices to allow the signals to arrive to at least one specified location in a specified way.

42. A method as in claim 41, wherein said specified way is that said signals are substantially synchronized with one another at said at least one specified location.

5 43. A method as in claim 41, wherein said specified way is that said signals are have a specified relationship to one another at said at least one specified location, which specified relationship is not synchronized, to allow a system to work properly.

10

44. A method as in claim 41, further comprising storing said information in a non-volatile memory.

ABSTRACT

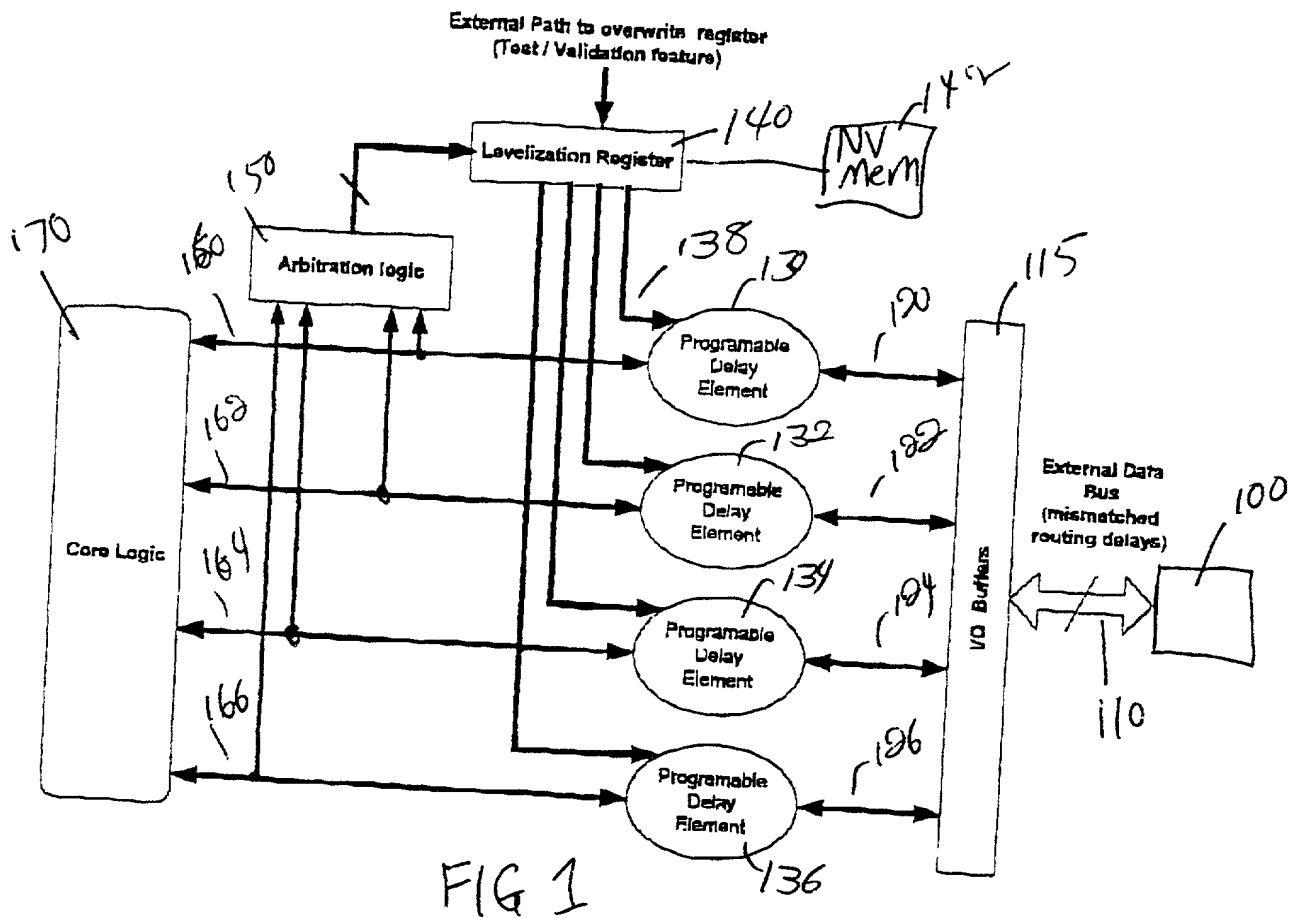
Technique and system for adjusting delays between
signals. A number of signals are produced, and delays between
the signals are determined. Programmable delay elements are
5 used, each driven by a signal indicative of one of the delays.
By delaying each of a number of the signals by different
amounts, the signals can be caused to arrive at desired times,
e.g., in synchronism with one another.

10

10052939.doc

005760-1502939

005T60-45029960



005T60" 73022960

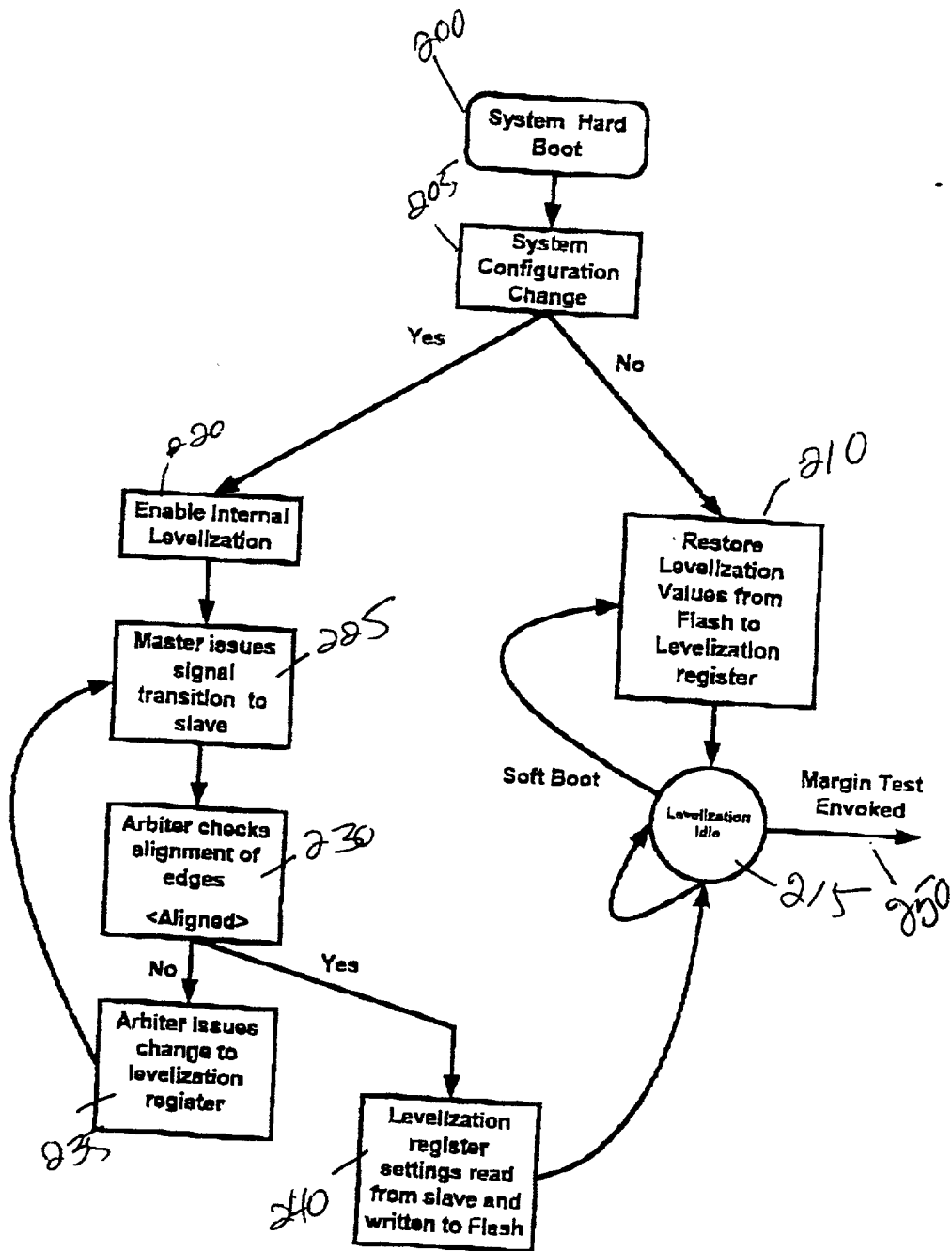


FIG 2

System Margin testing using line levelization registers.

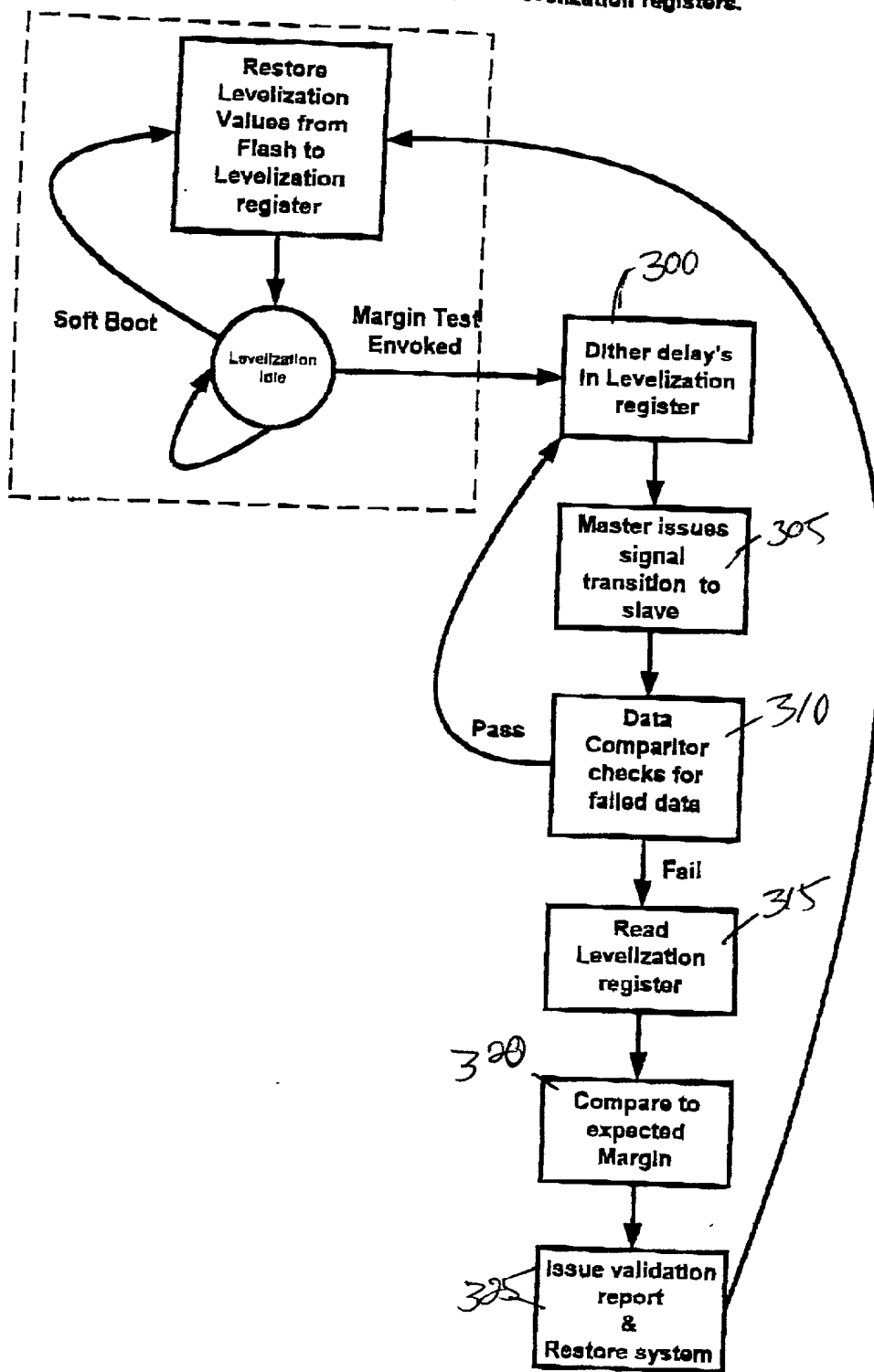


FIG 3